Amendments to the Claims:

This listing of claims replaces all prior versions and listings of claims in the application:

Listing of Claims:

1. (Currently amended) A method comprising:

generating a functional design of a logic circuit ~~by~~ based on selecting, placing, and connecting graphical elements ~~using~~ received from a graphical user interface (GUI), the graphical elements representing elements of the logic circuit, the elements comprising state elements, combinatorial logic, reusable library elements, and elements containing textual description of logic;

receiving an input through the GUI;

refining the functional design to represent a hardware design of the logic circuit using the input to modify the connectivity of the graphical elements;

maintaining a data structure representative of the functional design; and

generating an architectural, cycle-based simulation model of the functional design of the logic circuit and a separate implementation hardware description language (HDL) model of the hardware design of the logic circuit from the data structure, the implementation HDL model functionally equivalent to the architectural, cycle-based simulation model.

2. (Original) The method of claim 1 wherein the data structure comprises a description of a net list.

3. (Original) The method of claim 2 wherein the data structure comprises:

elements representing logical functions;

elements representing connection points to gates;

elements representing all bits of a simulation state; and

elements representing an arbitrary collection of bits within the simulation state.

4. (Previously presented) The method of claim 3 wherein the elements comprise C++ classes used to generate the architectural, cycle-based simulation model.

5. (Previously presented) The method of claim 1 wherein the architectural, cycle-based simulation model comprises C++ software code.

6. (Cancelled).

7. (Previously presented) The method of claim 1 wherein the HDL is Verilog.

8. (Previously presented) The method of claim 1 wherein the HDL is Very high speed integrated circuit Hardware Design Language (VHDL).

9. (Currently amended) A method comprising:

generating a model containing combinatorial blocks, state elements, reusable library elements, and elements containing textual description of logic ~~using~~ received from a graphical user interface (GUI), the model representing a functional design of a logic circuit;

receiving an input through the GUI;

refining the model to represent a hardware design of the logic circuit using the input;

maintaining a descriptive net list of the model; and

generating a C++ architectural, cycle-based simulation model and a functionally equivalent implementation Verilog model from the descriptive net list.

10. (Cancelled).

11. (Original) The method of claim 9 wherein the net list comprises gates, nodes and nets.

12. (Previously presented) The method of claim 9 wherein maintaining comprises parsing and analyzing the combinatorial blocks, state elements, reusable library elements, and elements containing textual description of logic of the model.

13. (Previously presented) The method of claim 9 wherein generating the C++ architectural, cycle-based simulation model and the functionally equivalent implementation Verilog model comprises:

partitioning a topology of the net list into a plurality of partitions; and

code ordering each of the partitions.

14. (Currently amended) A computer program product residing on a computer readable medium having instructions stored thereon which, when executed by the processor, cause the processor to:

generate a model containing combinatorial blocks, state elements, reusable library elements, and elements containing textual description of logic ~~using~~ received from a graphical user interface (GUI), the model representing a functional design of the logic circuit;

receive an input through the GUI;

refine the model with the input to represent a hardware design of the logic circuit;

maintain a descriptive net list of the model; and

generate a C++ architectural, cycle-based simulation model and a functionally equivalent implementation Verilog model from the descriptive net list.

15. (Original) The computer product of claim 14 wherein the computer readable medium is a random access memory (RAM).

16. (Original) The computer product of claim 14 wherein the computer readable medium is a read only memory (ROM).

17. (Original) The computer product of claim 14 wherein the computer readable medium is a hard disk drive.

18. (Currently amended) A processor and memory configured to:

generate a model containing combinatorial blocks, state elements, reusable library elements, and elements containing textual description of logic ~~using~~ received from a graphical user interface(GUI), the model representing a functional design of the logic circuit;

receive an input through the GUI;

refine the model from the input to represent a hardware design of the logic circuit;

maintain a descriptive net list of the model; and

generate a C++ architectural, cycle-based simulation model and a functionally equivalent implementation Verilog model from the descriptive net list.

19. (Original) The processor and memory of claim 18 wherein the processor and memory are incorporated into a personal computer.

20. (Original) The processor and memory of claim 18 wherein the processor and memory are incorporated into a network server residing in the Internet.

21. (Original) The processor and memory of claim 18 wherein the processor and memory are incorporated into a single board computer.

22. (Previously presented) A system comprising:

a graphic user interface (GUI) for receiving selections of graphical elements to generate a model and displaying the model, the model containing combinatorial blocks, state elements,

library elements, and elements containing textual description of logic, the model representing a functional, performance based model and a functionally equivalent implementation hardware design of a logic circuit;

a maintenance process to manage a data structure representing a descriptive net list of the model; and

a code generation process to generate a C++ architectural, cycle-based simulation model and a functionally equivalent implementation Verilog model from the data structure.

23. (Original) The system of claim 22 wherein the data structure comprises gates, nodes and nets.

24. (Previously presented) The system of claim 22 wherein the maintenance process comprises parsing and analyzing the combinatorial blocks, state elements, reusable library elements, and elements containing textual description of logic of the model.

25. (Original) The system of claim 22 wherein the code generation process comprises:

partitioning a topology of the net list into a plurality of partitions; and

code ordering each of the partitions.

26. (Withdrawn) A data structure comprising:

elements representing logical functions of a logic model;

elements representing connection points to gates of the logic model;

elements representing all bits of a simulation state of the logic model; and

elements representing an arbitrary collection of bits within the simulation state of the logic model.

27. (Withdrawn) The data structure of claim 26 wherein the elements are stored in a binary tree.

28. (Withdrawn) The data structure of claim 26 wherein the elements are stored in a linked list.